Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (Currently Amended) A method for automatically preventing errors in computer software having a plurality of different life cycle phases, the method comprising:

storing source code of the computer software in a code repository;

providing a plurality of software verification programs to verify the computer software, wherein each of the plurality of software verification programs relates to a respective lifecycle phase of the computer software[[,]] and automatically generates one or more test cases from the source code of the computer software, and wherein each of the plurality of software verification programs is an independent verification tool including a user interface and is capable of being executed in a batch process;

executing a first software verification program relating to a first lifecycle phase of the computer software, wherein the first software verification program automatically generates one or more test cases from the source code of the computer software;

executing a second software verification program relating to a ~~second~~ next lifecycle phase of the computer software different from the first lifecycle phase, wherein the second software verification program automatically generates one or more test cases from the source code of the computer software;

generating verification results for the first and the next ~~each respective~~ lifecycle phases ~~phase~~ of the computer software, responsive to executing the first and second software verification programs and the automatically generated test cases; and

processing the verification results for generating a representation of functional behavior of the computer software.

2. (Previously Presented) The method of claim 1 further comprising providing a common configuration file for the plurality of verification programs.

3. (Currently Amended) The method of claim 2, further comprising customizing a verification scope of one or more of the verification programs by modifying the common configuration file responsive to an objective criterion of quality of the computer software, wherein the objective criterion of quality is a quality rating of the entire computer software that takes into account the verification results of each of the verification programs, the number of test cases executed, and the success or failure of the test cases.

4. (Previously Presented) The method of claim 3 further comprising modifying a portion of the common configuration file specific to one of the plurality of verification programs responsive to the objective criterion of quality of the computer software.

5. (Previously Presented) The method of claim 3 further comprising modifying a portion of the common configuration file specific to one of a plurality of software developers responsive to the objective criterion of quality of the computer software.

6. (Previously Presented) The method of claim 1, further comprising formulating the verification results in a confidence factor represented by the equation:

$$C = p/t \times 100,$$

where p is number of successful test cases and t is total number of test cases.

7. (Original) The method of claim 1, wherein each portion of the computer software being developed by a software developer of a plurality of software developers, and the verification results include a plurality of objective criteria each of the plurality of objective criteria corresponding to quality of a respective portion of the computer software developed by each software developer of the plurality of software developers.

8. (Previously Presented) The method of claim 7 further comprising providing a common configuration file for the plurality of verification programs; and modifying the common configuration file responsive to one or more objective criteria corresponding to quality of a respective portion of the computer software developed by each software developer.

9. (Previously Presented) The method of claim 7 further comprising verifying a first portion of the computer software developed by a first developer of the plurality of software developers using the plurality of verification programs, before the computer software is stored in the code repository.

10. (Original) The method of claim 9 further comprising allowing storing the first portion of the computer software in the code repository only if result of verification of the first portion meets a set standard.

11. (Original) The method of claim 10 further comprising modifying the set standard responsive to the objective criterion of quality of the computer software.

12. (Original) The method of claim 10, wherein the set standard is common to each of the plurality of software developers.

13. (Original) The method of claim 10, wherein the set standard is unique to at least one of the plurality of software developers.

14. (Currently Amended) A system for automatically preventing errors in computer software having a plurality of different life cycle phases comprising:

a code repository for storing source code of the computer software;

means for providing a plurality of software verification programs to verify the computer software, wherein each of the plurality of software verification programs relates to a respective lifecycle phase of the computer software[[,]] and automatically generates one or more test cases from the source code of the computer software, and wherein each of the plurality of software verification programs is an independent verification tool including a user interface and is capable of being executed in a batch process;

means for executing a first software verification program relating to a first lifecycle phase of the computer software, wherein the first software verification program automatically generates one or more test cases from the source code of the computer software;

means for executing a second software verification program relating to a next ~~second~~ lifecycle phase of the computer software different from the first lifecycle phase, wherein the second software verification program automatically generates one or more test cases from the source code of the computer software;

means for generating verification results for the first and the next ~~each respective~~ lifecycle phases ~~phase~~ of the computer software, responsive to executing the first and second software verification programs and the automatically generated test cases; and

means for processing the verification results for generating a representation of functional behavior of the computer software.

15. (Previously Presented) The system of claim 14 further comprising means for providing a common configuration file for the plurality of verification programs.

16. (Currently Amended) The system of claim 15 further comprising means for modifying the common configuration file responsive to an objective criterion of quality of the computer software, wherein the objective criterion of quality is a quality rating of the entire computer software that takes into account the verification results of each of the verification programs, the number of test cases executed, and the success or failure of the test cases.

17. (Previously Presented) The system of claim 15 further comprising means for modifying a portion of the common configuration file specific to one of the plurality of verification programs responsive to an objective criterion of quality of the computer software.

18. (Previously Presented) The system of claim 15 further comprising means for modifying a portion of the common configuration file specific to one of a plurality of software developers responsive to an objective criterion of quality of the computer software.

19. (Previously Presented) The system of claim 14, further comprising means for formulating the verification results in a confidence factor represented by the equation:

$$C = p/t \times 100,$$

where p is number of successful test cases and t is total number of test cases.

20. (Original) The system of claim 14, wherein each portion of the computer software being developed by a software developer of a plurality of software developers, and the verification results include a plurality of objective criteria each of the plurality of objective criteria corresponding to quality of a respective portion of the computer software developed by each software developer of the plurality of software developers.

21. (Previously Presented) The system of claim 20 further comprising means for providing a common configuration file for the plurality of verification programs; and means for modifying the common configuration file responsive to one or more objective criteria corresponding to quality of a respective portion of the computer software developed by each software developer.

22. (Previously Presented) The system of claim 20 further comprising means for verifying a first portion of the computer software developed by a first developer of the plurality

of software developers using the plurality of verification programs, before the computer software is stored in the code repository.

23. (Original) The system of claim 22 further comprising means for allowing storing the first portion of the computer software in the code repository only if result of verification of the first portion meets a set standard.

24. (Original) The system of claim 23 further comprising means for modifying the set standard responsive to the objective criterion of quality of the computer software.

25. (Original) The system of claim 23, wherein the set standard is common to each of the plurality of software developers.

26. (Original) The system of claim 23, wherein the set standard is unique to at least one of the plurality of software developers.

27. (Currently Amended) A method for automatically preventing errors in computer software having a plurality of different life cycle phases, the method comprising:

~~providing a known~~ detecting an error in the computer software, the detected ~~known~~ error belonging to a class of errors;

providing a plurality of software verification programs each of the plurality of software verification programs related to a respective lifecycle phase of the computer software, wherein each of the plurality of software verification programs is an independent verification tool including a user interface and is capable of being executed in a batch process;

analyzing the detected ~~known~~ error in the computer software;

determining what phase of the lifecycle the detected error was introduced, based on analyzing the detected ~~known~~ error;

customizing a verification scope of one or more of the plurality of verification programs that correspond to the determined lifecycle phase wherein the detected ~~known~~ error was introduced; and

executing the plurality of software verification programs to verify the class of the detected ~~known~~ error is detected in a ~~respective~~ same lifecycle phase of the computer software as the determined lifecycle phase wherein the detected error was introduced.

28. (Currently Amended) The method of claim 27, further comprising customizing the verification scope by modifying a configuration file common to the verification programs based on an objective criterion of quality of the computer software, wherein the objective criterion of quality is a quality rating of the entire computer software that takes into account verification results of each of the verification programs, the number of test cases executed, and the success or failure of the test cases.

29. (Previously Presented) The method of claim 28 further comprising modifying a portion of the configuration file specific to one of the plurality of verification programs based on the objective criterion of quality of the computer software.

30. (Original) The method of claim 28 further comprising modifying a portion of the common configuration file specific to one of a plurality of software developers responsive to the objective criterion of quality of the computer software.

31. (Previously Presented) The method of claim 27, further comprising processing the verification results for generating an objective criterion of quality of the computer software by formulating the verification results in a confidence factor represented by the equation:

$$C = p/t \times 100,$$

where p is number of successful test cases and t is total number of test cases.

32. - 42 (Canceled)

43. (Previously Presented) The method of claim 28 further comprising customizing the verification scope of one or more of the plurality of verification programs for a second time, if the known error is not detected by executing the plurality of software verification programs.

44. (Previously Presented) The method of claim 27 further comprising executing the plurality of software verification programs periodically to prevent the known error from re-occurring when the computer software is modified.

45. (Currently Amended) A system for automatically preventing errors in computer software having a plurality of different life cycle phases comprising:

means for detecting an ~~providing a known~~ error in the computer software, the detected ~~known~~ error belonging to a class of errors;

means for providing a plurality of software verification programs each of the plurality of software verification programs related to a respective lifecycle phase of the computer software;

means for analyzing the detected ~~known~~ error in the computer software;

means for determining what phase of the lifecycle the detected error was introduced, based on analyzing the detected ~~known~~ error;

means for customizing a verification scope of one or more of the plurality of verification programs that correspond to the determined lifecycle phase wherein the detected ~~known~~ error was introduced; and

means for executing the plurality of software verification programs to verify the class of the detected ~~known~~ error is detected in a ~~respective~~ same lifecycle phase of the computer software as the determined lifecycle phase wherein the detected error was introduced.

46. (Previously Presented) The system of claim 45 further comprising means for executing the plurality of software verification programs to verify the known error is detected in computer software.

47. (Previously Presented) The system of claim 46 further comprising means for customizing the verification scope of one or more of the plurality of verification programs for a second time, if the known error is not detected by executing the plurality of software verification programs.

48. (Previously Presented) The system of claim 45 further comprising means for executing the plurality of software verification programs periodically to prevent the known error from re-occurring when the computer software is modified.